IN THE CLAIMS:

This listing of claims will replace all prior versions, and listing of claims, in the Application.

Listing of claims:

1. (Currently amended) A method of synchronizing ~~at least two~~ N concurrently running processes in a data processing system at a first phase before allowing the N processes to proceed to a second phase, N >= 2, comprising:

~~(a)~~ providing a first array of N elements ~~with~~ initialized each to a first state ~~states,~~ wherein each i^{th} element of said first array is associated with an i^{th} ~~having a~~ concurrently running process ~~associated therewith, each element of said first array being configured to have its state updated~~ which will update the i^{th} element of the first array to a second state ~~by its associated concurrently running process upon completion of a~~ in response to completing the first phase ~~by said associated concurrently running process~~, where 1 = i <= N;

~~(b)~~ providing a second array of N elements ~~with~~ initialized each to a hold state ~~states,~~ wherein each i^{th} element of said second array is associated with the i^{th} ~~having a~~ concurrently running process associated with the i^{th} element of the first array ~~therewith, each element of said second array~~ and ~~being configured~~ is used to hold the i^{th} associated concurrently running process at the first phase, and is enabled to switch, in response to ~~upon~~ receiving [[an]] a release instruction, to a release state to release the i^{th} associated concurrently running process to proceed to the second phase; and

~~(c)~~ using a designated process configured to ascertain when the N elements of the first array are updated to the second state to issue the release instruction to allow the N processes to proceed to the second phase ~~arranging for monitoring said first array of elements and, upon each element of said first array having had its state updated, arranging for generating said instruction for switching said elements of said second array to said release state~~.

2. (Currently amended) The method recited in claim 1, ~~further comprising:~~

~~(d)~~ wherein ~~for~~ each i^{th} process of said N ~~at least two~~ concurrently running processes~~, configuring said each process such that, upon completion of said phase and upon updating of its associated element of said first array, said each process then~~ waits at its i^{th} associated element of said second array for said release state in response to completing the first phase and updating the i^{th} associated element of the first array.

3. (Currently amended) The method recited in claim 2, wherein each i^{th} element of said first array has a byte size corresponding to ~~the~~ a size of a cache line used in said data processing system.

4. (Currently amended) The method recited in claim 3, wherein each i^{th} element of said second array has a byte size corresponding to the size of said cache line used in said data processing system.

5. (Currently amended) The method recited in claim 4, ~~further comprising providing~~ wherein each i^{th} element of said second array is provided locally in relation to its i^{th} respective, associated process.

6. (Currently amended) The method recited in claim 2, ~~further comprising,~~ wherein after the N elements ~~upon said each element~~ of said first array ~~having had its state~~ are updated to the second state, and prior to ~~generating said~~ issuance of the release instruction ~~for switching said elements of said second array to said release state, arranging for reinitializing each element~~ the N elements of said first array are reinitialized to the first state.

7. (Currently amended) The method recited in claim 1, wherein ~~in (c), said monitoring of said first array of elements is performed by~~ the designated process is one of said N concurrently running processes.

8. (Currently amended) The method recited in claim 1, wherein ~~in (c), said monitoring of said first array of elements is performed by~~ the designated process is not one of the N concurrently running processes ~~an independent process~~.

9. Canceled.

10. (Currently amended) The method recited in claim 1, wherein ~~in (a), said~~ each i^{th} element of said first array and said second array comprises a state machine.

11. (Original) The method recited in claim 10, wherein said state machine is one of a counter, a gate, a flag and a sensor.

12. Canceled.

13. Canceled.

14. (Currently amended) A system having at least one processor for processing instructions to synchronize ~~for synchronizing at least two~~ N concurrently running processes ~~in a data processing system~~ at a first phase before allowing the N processes to proceed to a second phase, where N >= 2, the instructions[[,]] comprising instructions to use:

~~(a)~~ a first array of N elements initialized each to a first state, each i^{th} element of said first array having [[a]] an i^{th} concurrently running process associated therewith[[,]]~~said each element of said first array being configured to have an initial state that may be updated by its~~ which will update the i^{th} element of the first array to a second state in response to completing the first phase, where 1 = i <= N ~~associated concurrently running process, upon completion of a phase by said associated concurrently running process~~;

~~(b)~~ a second array of N elements with each element initialized to a hold state wherein each i^{th} element of said second array is associated with the i^{th} ~~having a~~ concurrently running process associated with the i^{th} element of the first array ~~associated therewith, each element of said second array being configured~~ and is used to hold the i^{th} associated concurrently running process at the first phase, and is enabled to switch, in response to receiving a release instruction, to a release state to release the i^{th} associated concurrently running process to proceed to the second phase ~~have an initial hold state that may be switched, upon receiving an instruction, to a release state~~;

~~(c)~~ a designated process configured to ascertain when the N elements of the first array are updated to the second state to issue the

release instruction to allow the N processes to proceed to the second phase ~~monitoring process for monitoring said first array of elements, said monitoring process being configured to generate said instruction for switching said elements of said second array to said release state, upon each element of said first array having had its state updated~~.

15. (Currently amended) The system recited in claim 14, wherein each i^{th} element of said first array has a byte size corresponding to ~~the~~ a size of a cache line used in said data processing system.

16. (Currently amended) The system recited in claim 15, wherein each i^{th} element of said second array has a byte size corresponding to the size of said cache line used in said data processing system.

17. (Currently amended) The system recited in claim 14, wherein each i^{th} element of said second array is provided locally in relation to its respective, i^{th} associated process.

18. (Currently amended) The system recited in claim 14, wherein ~~said~~ each i^{th} element of said first array and said second array is a state machine.

19. (Currently amended) The system recited in claim 14, wherein said state machine is ~~each element in~~ one of a counter, a gate, a flag and a switch.

20. Canceled.

21. Canceled.

22. (Currently amended) The system recited in claim 14, wherein said ~~at least two~~ N concurrently running processes execute on multiple processors embodied within a single computer.

23. (Currently amended) The system recited in claim 14, wherein said ~~at least two~~ N concurrently running processes execute on multiple processors distributed across multiple computers connect across a network.

24. (Currently amended) A processor for executing a process in order to synchronize said process at a phase with at least one other concurrently running process, said processor being operable to:

access elements of a first array of elements, each accessed element of the first array being associated with one concurrently running process completing the phase and having an initial state;

~~access an element of a first array of elements, said element of said first array being associated with said process, said element of said first array having an initial state~~;

update an accessed ~~said~~ element of said first array of elements to another state upon completion of [[a]] the phase by said process, the accessed element being associated with the process;

~~after said updating,~~ access an element of a second array of elements, said element of said second array being associated with said process[[,]] wherein all elements ~~said element~~ of said second array ~~having~~ have an initial hold state to hold an associated process at the phase and ~~being~~ are enabled ~~configured~~ to switch, in response to ~~upon~~ receiving an instruction,

to a release state[[,]] to release the associated process to proceed to a next phase; and

continually check ~~said element of~~ said ~~second~~ first array, using a designated process enabled to do so, to determine when all elements of the first array are updated in order to issue the instruction ~~for said switch to said release state until detecting said release state~~.

25. (Currently amended) A method for executing a process in order to synchronize said process at a phase with at least one other concurrently running process, comprising:

accessing elements of a first array of elements, each accessed element of the first array being associated with one concurrently running process completing the phase and having an initial state;

~~accessing an element of a first array of elements, said element of said first array being associated with said process, said element of said first array having an initial state~~;

updating an accessed ~~said~~ element of said first array of elements to another state upon completion of [[a]] the phase by said process, the accessed element being associated with the process;

~~after said updating,~~ accessing an element of a second array of elements, said element of said second array being associated with said process[[,]] wherein all elements ~~said element~~ of said second array ~~having~~ have an initial hold state to hold an associated process at the phase and ~~being~~ are enabled ~~configured~~ to switch, in response to ~~upon~~ receiving an instruction,

to a release state[[,]] to release the associated process to proceed to a next phase; and

continually checking ~~said element of~~ said ~~second~~ first array, using a designated process enabled to do so, to determine when all elements of the first array are updated to the other state in order to issue the instruction ~~for said switch to said release state until detecting said release state~~.

26. (Currently amended) A processor for executing a designated process in order to synchronize at least two concurrently running processes at a phase, said processor being operable to:

access, using the designated process, an ~~a first~~ array of elements, each element of said ~~first~~ array of elements being associated with one of said at least two concurrently running ~~process~~ processes and having an initial state;

monitor, using the designated process, all elements of said ~~first~~ array of elements to detect when ~~until detecting that~~ each of said elements of said ~~first~~ array has been updated by its associated process;

hold, using the designated process, each one of the at least two concurrently running processes at the phase until an instruction to release the at least two concurrently running processes is received, the instruction indicating that the at least two concurrently running processes are synchronized at the phase; and

~~thereafter~~ generate [[an]] the instruction, using the designated process, when it is determined that all the elements of the array have been updated

~~to switch all elements of a second array of elements from an initial hold state to a release state, each element of said second array of elements being associated with one of said at least two concurrently running processes~~.

27. (Currently amended) The processor recited in claim 26, wherein said designated process ~~executed thereon~~ is one of said concurrent processes.

28. (Currently amended) A computer program product for synchronizing ~~at least two~~ N concurrently running processes in a data processing system at a first phase before allowing the N processes to proceed to a second phase, N >= 2, the computer program product comprising:

a computer useable medium having computer readable program code means embodied in the medium for synchronizing the N ~~at least two~~ concurrently running processes, the computer program code means including:

computer readable program code means for providing a first array of N elements ~~with~~ initialized each to a first state ~~states~~, each i^{th} element of said first array being associated with an i^{th} ~~having a~~ concurrently running process which will update the i^{th} element of the first array to a second state in response to completing the first phase, where 1 = i <= N ~~associated therewith, each element of said first array and being configured to have its state updated by its associated concurrently running process upon completion of [[a]] phase by said associated concurrently running process~~;

computer readable program code means for providing a second array of N elements ~~with~~ initialized each to a hold state ~~states~~, each

i^{th} element of said second array being associated with the i^{th} ~~having~~ ~~a~~ concurrently running process associated with the i^{th} element of the first array and is enabled ~~therewith, each element of said second array being configured~~ to hold the i^{th} associated concurrently running process at the first phase and to switch, in response to ~~upon~~ receiving [[an]] a release instruction, to a release state to release the i^{th} associated concurrently running process to proceed to the second phase; and

computer readable program code means for monitoring said first array of N elements and, in response to ~~upon~~ each i^{th} element of said first array having had its state updated, generating said instruction for switching said N elements of said second array to said release state to allow the N processes to proceed to the second phase, said computer readable program monitoring code means including computer readable program code means for processing a designated process to monitor the first array of N elements and issue the generated instruction.

29. (Currently amended) The computer program product recited in claim 28, ~~further comprising:~~

wherein ~~computer readable program code means for configuring~~ each i^{th} process of said ~~at least two~~ N ~~at least two~~ concurrently running processes~~,~~ ~~upon completion of said phase and upon updating of its associated element of said first array, to wait~~ waits at its i^{th} associated element of said second array for said release state in response to completing the first phase and updating the i^{th} associated element of the first array.

30. (Currently amended) A system for synchronizing ~~at least two~~ N concurrently running processes ~~in a data processing system~~ at a first phase before allowing the N processes to proceed to a second phase, N >= 2, comprising:

~~(a)~~ means for providing a first array of N elements ~~with~~ initialized each to a first state ~~states,~~ each i^{th} element of said first array being associated with an i^{th} ~~having a~~ concurrently running process ~~associated therewith, each element of said first array~~ which will update the i^{th} associated element of the first array to a second state in response to completing the first phase, where 1 = i <= N ~~being configured to have its state updated by its associated concurrently running process upon completion of a phase by said associated concurrently running process~~;

~~(b)~~ means for providing a second array of N elements ~~with~~ initialized each to a hold state ~~states~~, each i^{th} element of said second array being associated with the i^{th} ~~having a~~ concurrently running process associated with the i^{th} element of the first array and is enabled ~~therewith, each element of said second array being configured~~ to hold the i^{th} associated concurrently running process at the first phase and is enabled to switch, in response to ~~upon~~ receiving [[an]] a release instruction, to a release state to release the i^{th} associated concurrently running process to proceed to the second phase; and

~~(c)~~ means for monitoring said first array of N elements and, upon each i^{th} element of said first array having had its state updated, generating said instruction for switching said N elements of said second array to said release state to allow the N processes to proceed to the second phase, said monitoring means including a

process designated to monitor the first array of N elements to determine when the N elements of the first array are updated to the second state and to issue the generated instruction when processed by a processor.

31. (Currently amended) The system recited in claim 30, ~~further comprising:~~

~~(d)~~ wherein ~~means for~~ each i^{th} process of said N ~~at least two~~ concurrently running processes~~, configuring said each process such that, upon completion of said phase and upon updating of its associated element of said first array, said each process then~~ waits at its i^{th} associated element of said second array for said release state in response to completing the first phase and updating the i^{th} associated element of the first array.